Introduction to Machine Learning - Age recognition project

Weronika Plichta, Szymon Trochimiak, Bartosz Grabek, Filip Kucia, Michał Taczała

January 26, 2024

1 Introduction

Age prediction based on face images is one of the many challenges in Computer Vision and Artificial Intelligence. Extracting relevant features from the image data is a nontrivial task. This report describes the research and development of Age Detect App, a desktop application for predicting age based on images. In the next sections we describe the process of data collection, analysis, age prediction model design, training and evaluation. The last section is about application setup and usage.

1.1 Our main responsibilities

- Data Collection - Filip Kucia
- Face Detection - Michał Taczała
- GUI and Integration - Bartosz Grabek
- Yolo v8 Model - Weronika Plichta
- ResNet Model - Szymon Trochimiak

1.2 Background knowledge

To successfully complete age detection, some prior knowledge was needed, including:

- **Machine Learning Fundamentals** - It consists of basic concepts like supervised/unsupervised learning. Moreover, understanding machine learning models like regression and classification was crucial.
- **Python programming** - The entire project was created using Python and pandas library for handling data. Using Python for machine learning-oriented tasks is a standard solution, so the knowledge of syntax and libraries was important.
- **Evaluation metrics** - There are many evaluation metrics like accuracy, precision, recall, and so on. In order to compare a few models with each other having a correctly chosen metric is the most important step
- **Model deployment** - Docker is the tool that we used to be sure, that our work can be run by everyone. Many things depend on the operating system and its settings, so we chose to have a standardized environment that will be universal for every user.
- **Data choosing** - To choose good data for this task we had to know the difference between labeled and unlabeled data. Moreover, data from different sources had errors and not finding them could cause wrong predictions for a model.
- **Balancing the dataset** - Because there were a larger number of people of a certain age, the knowledge of techniques like under-sampling or over-sampling was needed to prevent a model from being biased towards a certain age group or ethnicity. There were situations, where there were a few people of a certain ethnicity, and we had to look for another dataset that could fill in that gap.
- **Data analysis** - To retrieve some information from the data, the knowledge from distributions, plotting, and missing values was needed.
- **Image processing** - In order to obtain the cropped face images, image processing skills were important.

2 Datasets

For the project, a combination of several comprehensive and diverse datasets was employed, each contributing unique attributes to enhance the accuracy and robustness of the age detection solution.

2.1 KANFace

KANFace[2] is an extensive dataset, comprising of 40,000 still images, is particularly tailored for facial analysis tasks, including age detection. The images in KANFace are curated to provide a wide range of facial expressions, angles, and lighting conditions, thereby enriching the dataset's utility in real-world scenarios. This diversity is crucial in training models to recognize and accurately predict ages under various imaging conditions.

2.2 UTKFace

UTKFace[6] is a standout dataset in terms of its size and age span, UTKFace includes over 20,000 images covering an age range from newborns to centenarians (0 to 116 years). This extensive age range is complemented by annotations for age, gender, and ethnicity. The inclusion of these multi-dimensional labels makes UTKFace an invaluable resource for developing a nuanced age detection model that is sensitive to the subtleties of aging across different genders and ethnicities.

2.3 IMDB-WIKI

With its impressive collection of over 500,000 face images labelled with age and gender, IMDB-WIKI[4, 5] is unparalleled in its scale. Derived from public domains like IMDB and Wikipedia, it offers a broad spectrum of facial features, skin tones, and expressions from various ethnic backgrounds and age groups. This extensive variety helps in training more robust and universally applicable age detection models.

2.4 AgeDB

The Age Database (AgeDB)[3] adds another layer of diversity. Accessible through AgeDB, it provides a well-annotated set of images specifically focused on age-related facial features. The dataset's structure and annotations are particularly useful for refining age detection algorithms to identify subtle age-related changes in facial features.

2.5 All-Age-Faces (AAF)

All-Age-Faces (AAF)[1], with its 13,322 face images predominantly of Asian origin, addresses the need for ethnic diversity in age detection systems. The age distribution ranges from 2 to 80 years, with a balanced representation of both genders (7,381 females and 5,941 males). The AAF dataset's focus on Asian faces fills a critical gap often overlooked in facial analysis datasets, providing valuable data for training age detection models that are effective across different ethnicities.

2.6 Comments on the data

The incorporation of these datasets into the human age prediction project ensures that the developed solution is not only versatile and capable of operating across a wide range of conditions but also sensitive to the variations in aging processes across different ethnicities and genders. This comprehensive approach, leveraging the strengths of each dataset, significantly enhances the performance and applicability of the age detection models in real-world scenarios.

3 Data analysis

First, we checked the distribution of the data (Figure 1). We can see that the distribution has very long (and unrealistic) tails.

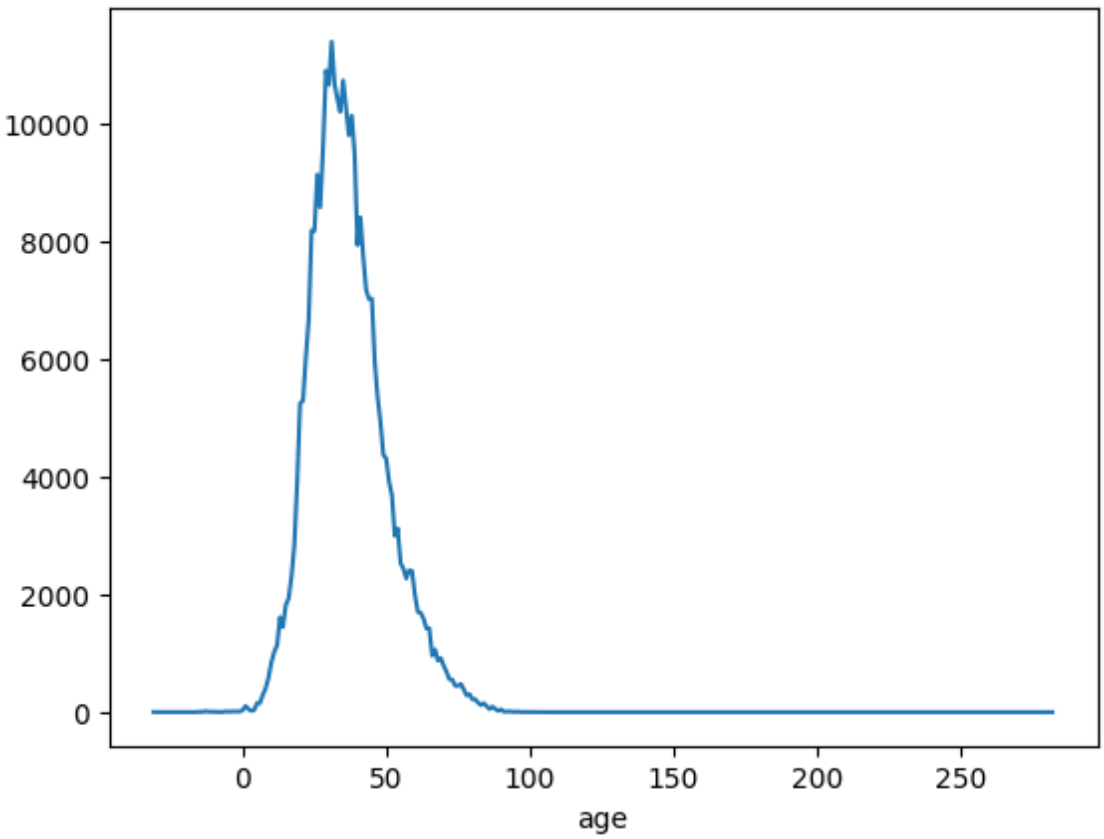

Figure 1: Original data distribution

We decided to only include the data between age 5 and 85 years. We also ran our face detection algorithm on the whole dataset to ensure that there is exactly one face per image, the face fills the whole image and is at least 40x40 pixels. The dataset was split into 3 sets: training (71 197 examples - 80% of the original data), validation (1 893 examples - 10%), test (1 635 examples - 10%). After each epoch, we will be validating the model on the validation dataset. However, the test dataset is only used for final evaluation. The model does not use any of the test data during the training process. Here are the final distributions of training (Figure 2), validation (Figure 3), and test (Figure 4).

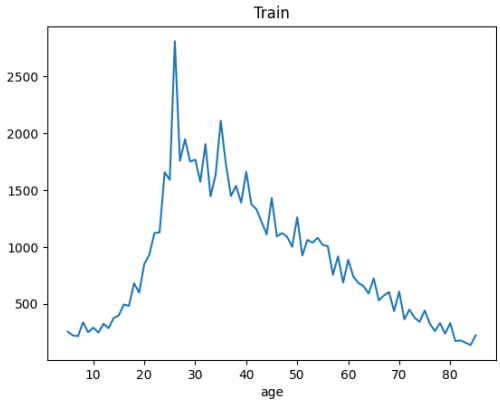

Figure 2: Train data distribution

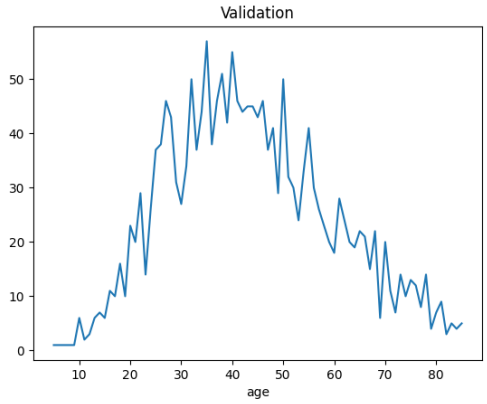

Figure 3: Validation data distribution

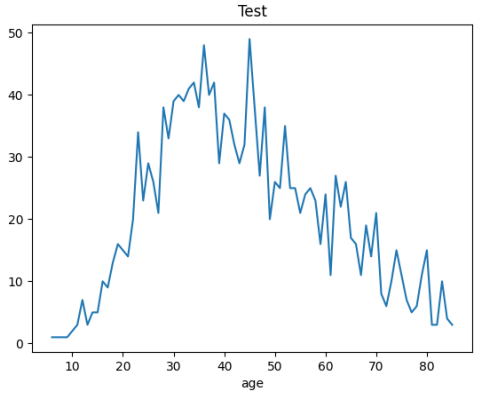

Figure 4: Test data distribution

4 Evaluation of the Face Recognition algorithm

4.1 Introduction

The face recognition algorithm is a vital component of our pipeline. To facilitate the successful operation of our age prediction model, it is essential to provide it with cropped faces. The age prediction model lacks the inherent capability to crop images, and failure to do so may result in the model learning from various elements other than faces.

Additionally, there are instances where a single image contains multiple faces. In such cases, accurate face detection is crucial as our age recognition model assumes a singular face in the photo. Incorrect face detection could lead to inaccurate age predictions.

Finally, after the research, we decided to use either the mediapipe or the opencv2 one.

4.2 Methodology

4.2.1 Dataset

The algorithms that we chose to implement are quite popular and went through many different kinds of testing from many people all over the world. Despite that, we've tested them ourselves, using our personal photos, and IMDB photos for reference.

4.3 Evaluation metrics

- **Accuracy** - The proportion of True positives and to total number of faces.
- **Speed** - The average time required to detect faces in the images.
- **Precision** - The proportion of True positives to the sum of True positives and False positives

4.4 Results

Both algorithms were tested on 81 randomly selected photos (both algorithms tested on the same set). There were about 130 faces, most of them of small size and almost recognizable, and that's the main reason for the low accuracy results.

	Speed [s]	Precision[%]	Accuracy[%]
OpenCV	28	95	25
Mediapipe	4	89	53

4.5 Detected faces - example

Figure 5: Faces detected using Mediapipe

4.6 Special case scenarios

Individuals seize numerous images in various settings. It is affordable to count on that the majority of those pictures deviate from the suitable picture required for an ID card, often proposing faces that can be both blurred or distorted. Consequently, the expectancy of flawless performance from our face detection model in every state of affairs is unrealistic, and, certainly, this is frequently contemplated in practical programs. In this section, we are able to discover instances of unconventional and unusual detections that stand up because of the inherent challenges posed by using numerous photographic situations

Figure 6: No faces detected

Figure 7: False faces detected by Mediapipe

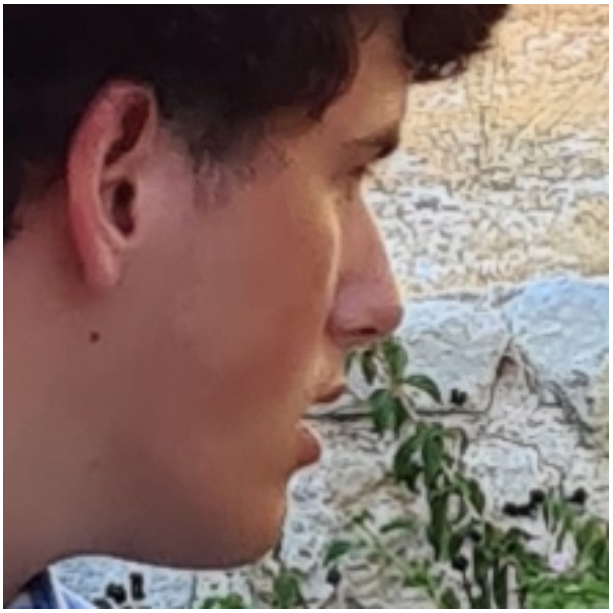

Figure 8: Face detected by Mediapipe and not detected by OpenCV

4.7 Conclusion

Mediapipe outperforms OpenCV in all scenarios tested, but it's not the perfect tool either. While both models work quite well for simple, front-facing images, Mediapipe excels in handling rotated or smaller face. However, any of these algorithms performs well with detecting really small faces. Ideally, the face should be as big as it's possible. If it's not, then we can't be sure whether it is recognized or not. Mediapipe model demonstrates significantly faster processing time, up to 7x than OpenCV. It happens that the images detected by the models are not actually faces, but these situations happen quite rarely. Despite the superior performance of the Mediapipe face detector, we opted to implement both solutions for a user to have a choice, and for comparison during testing.

5 ResNet50

5.1 Introduction

ResNet50 is a popular image classification. It's known for its efficiency and accuracy and has been trained on the ImageNet dataset.

We are taking the pre-trained ResNet50 model to use it for making a regression algorithm that predicts the age of a person. The motivation is that the pre-trained feature extractors (convolution layers), which were also trained on faces, are better suited for extracting relevant facial features for age estimation than untrained layers.

We are only using the convolution layers with added two fully connected layers. The intermediate FC layer has 128 neurons with ReLU activation function, and the final layer has 1 neuron with no activation function.

5.2 Data balancing

Due to the imbalances in the training data, we decided to make the data more uniform across all selected ages. To accomplish this, we wrote a custom data extraction function that calculates the number of examples for each class and we take the minimal value (called m). In each epoch, we randomly take exactly m examples from each class. If there are more than m examples in a given class, then they are spread across multiple epochs.

This approach ensures that our trained model does not try to replicate the distribution of the data, thus reducing the bias.

5.3 Augmentations

Our base can be seen in Figure 9 (with image resized to 224x224 and learning rate of 0.001). It's a bit unstable, overfits fast, and stops learning after a while.

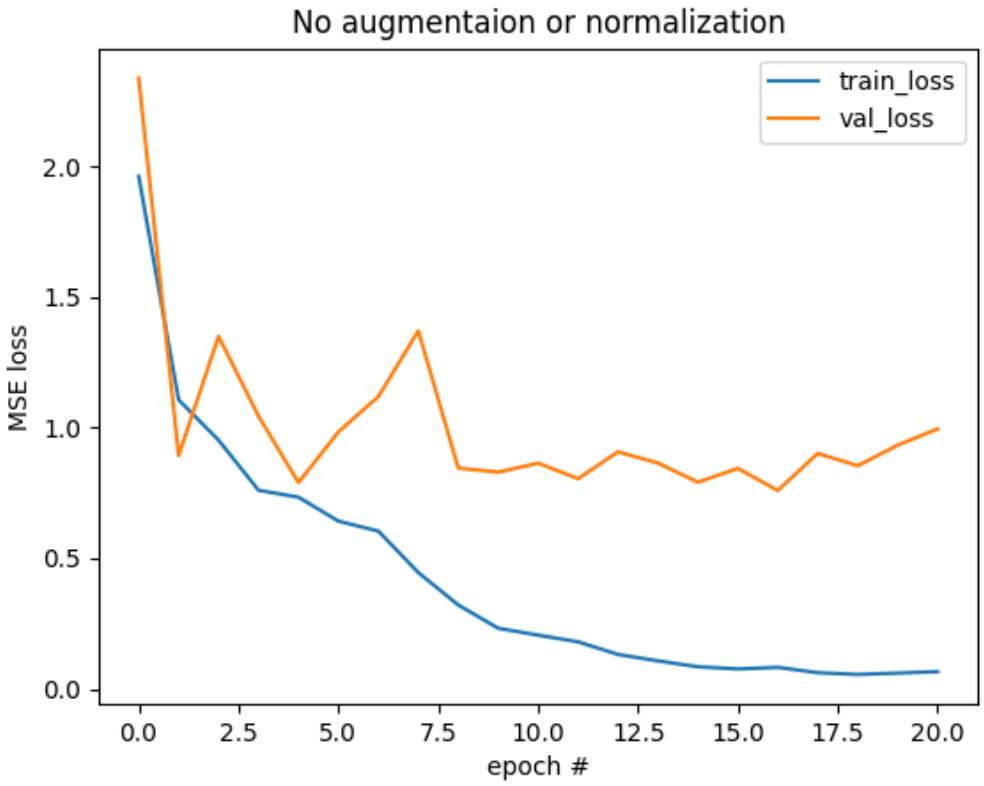

Figure 9: No augmentation

We normalized the input values by subtracting the mean of colors averaged across the training dataset (0.5646, 0.4326, 0.3711) and then dividing by the standard deviation (calculated in the same way) (0.2495, 0.2205, 0.2173). We also added a dropout between the FC layers and set it to 0.2. Finally, we lowered the learning rate to 0.0001. We can see the results in Figure 10. The loss is lower than in the base case and overfitting doesn't occur for longer time.

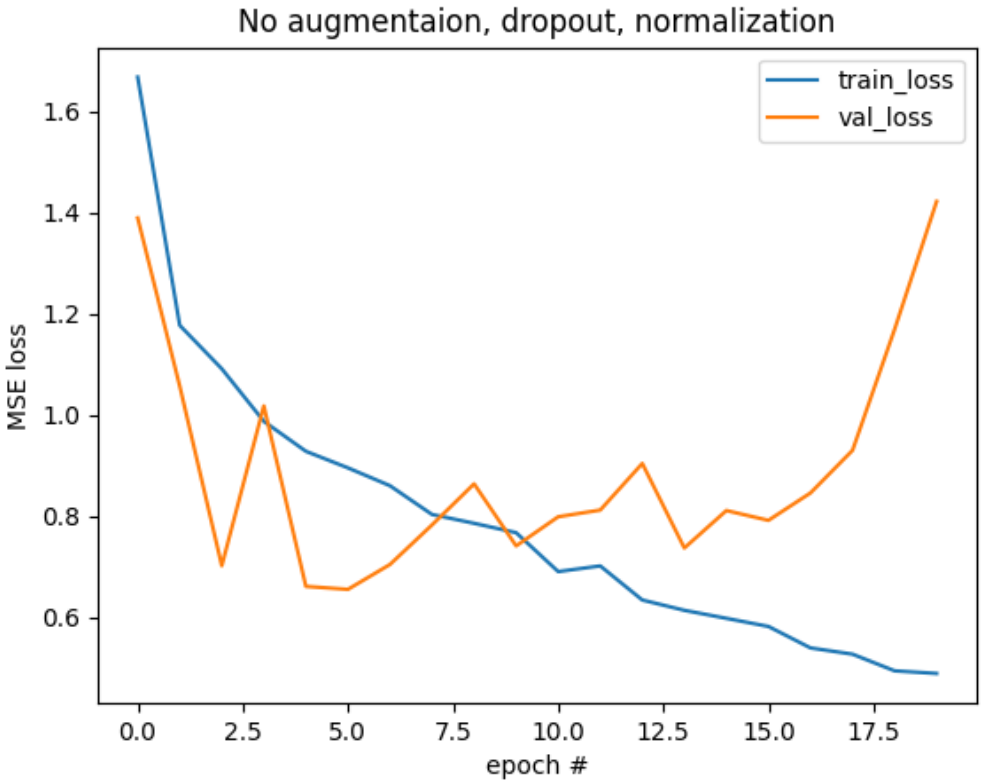

Figure 10: Dropout and Normalization

Finally, we added flip, rotation, color augmentations and gaussian noise. In Figure 11, we can see that the model doesn't overfit and has better learning stability.

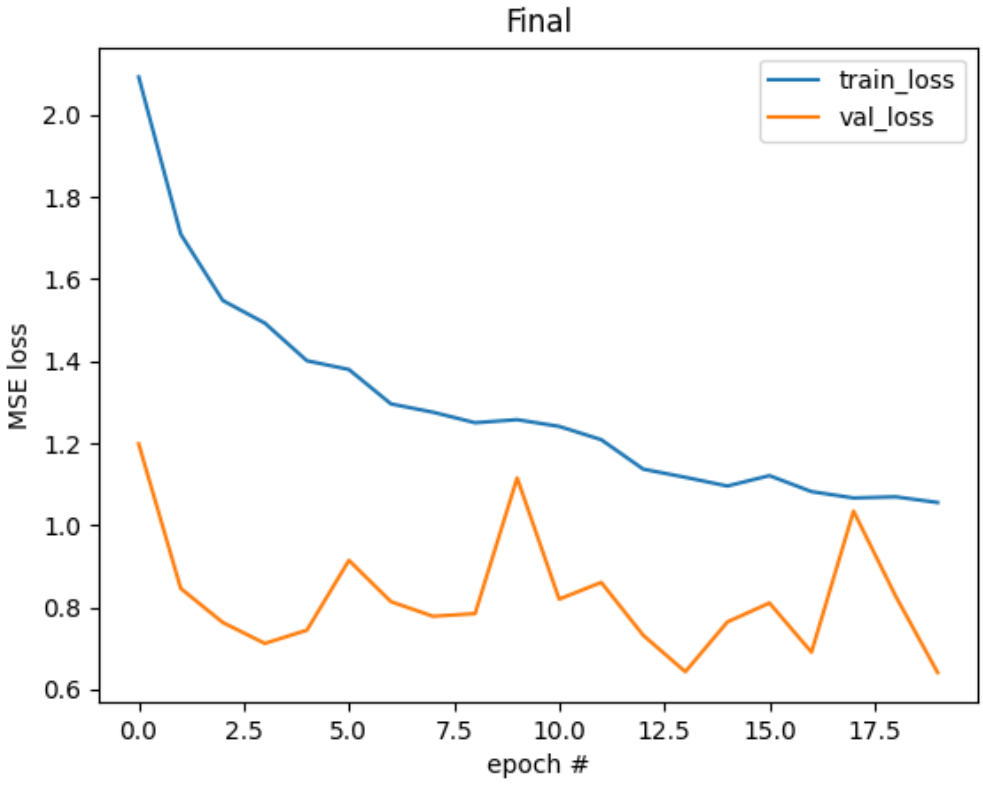

Figure 11: Final model

5.4 Final results

The final model was trained for 20 epochs with these parameters:

- learning rate: 0.0001
- batch size: 256
- normalized input and output
- dropout: 0.2
- flip, rotation and color augmentations
- gaussian noise with std=0.2

The learning progress was visualized in Figure 7. The results for test dataset:

Mean Squared Error	Average age error (in years)
0.67	**9.01**

6 YOLOv8 classifier

6.1 Introduction

YOLOv8 was introduced at the beginning of 2023, as the next iteration of popular, state of the art YOLO (You Only Look Once). The most recognizable utility of this model is detection of objects. YOLOv8 introduced pretrained classification model. It was pretrained on one of the biggest dataset of images - ImageNET. With that we want to test how it will handle age classification task.

6.2 Methodology

Model was trained on the combination of all of the datasets that were mentioned previously. Every image had a size of 224x224. 8 classes were defined. We've started from 5 years old, as data for 1-4 years old was very poor and imprecise.

- 5-14
- 15-24
- 25-34
- 35-44
- 45-54
- 55-64
- 65-74
- 75-84

Training was done for 20 epochs, next for 25 epochs, however with increasing number of epochs, we didn't see any significant improvement we stopped adding more epochs. The accuracy was better with decreasing number of classes.

6.3 Results

Model did not live up to the expectations and showed poor results. Below are the accuracy plots with the peak reaching only 46%.

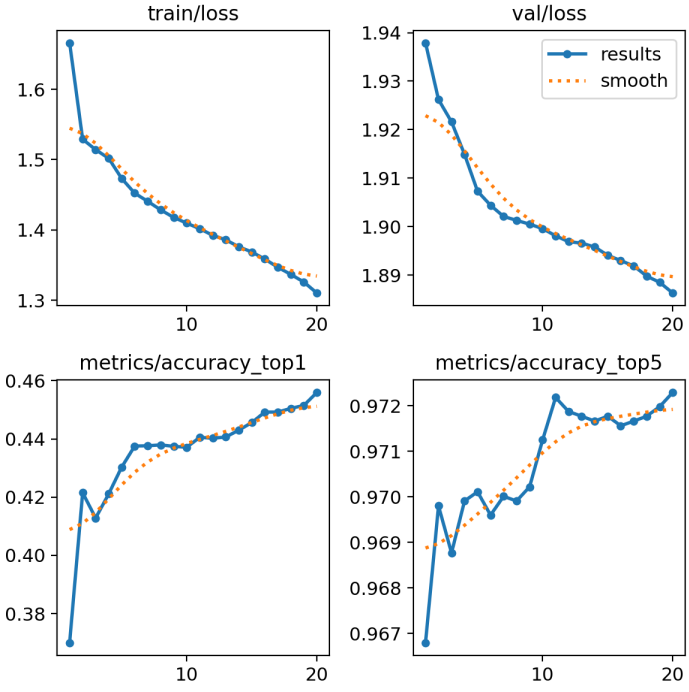

Figure 12: Accuracy and loss graphs

On the following confusion matrix, we can observe that the best accuracy is obtained in 25-34 age range:

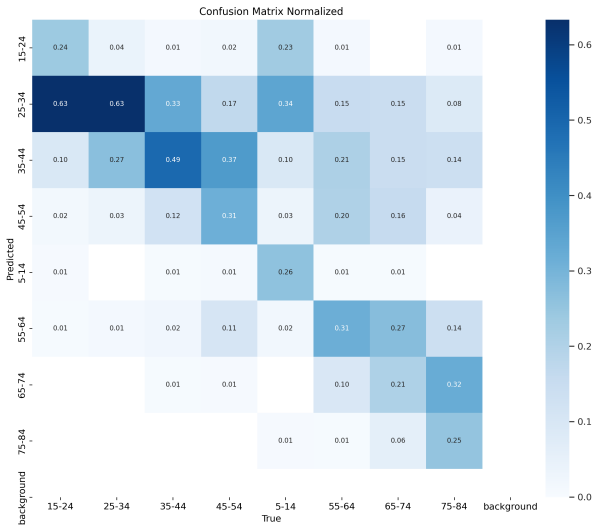

Figure 13: Yolov8 confusion matrix

One of the reasons could be underrepresentation of images in higher age ranges. The main aim of the YOLO models is object detection, rather than classification task which in this case is under performing.

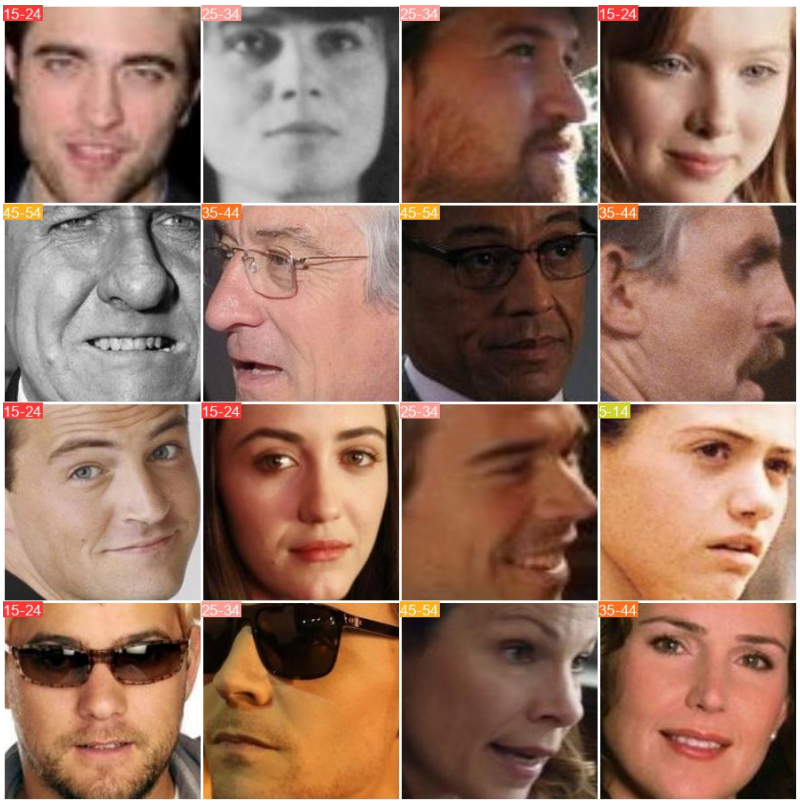

Figure 14: Age classification

7 Age Detection App - User Manual

7.1 Links to resources

Image datasets and trained models can be found here: Collected data
All created files can be found here: Github repository

7.2 Setup

7.2.1 Prerequisites

Make sure you have the following installed on your system:

- `Python 3.11+`
- `pip (Python package installer)`

7.2.2 Clone the repository

```
git clone https://github.com/Bartosz7/age-detect-app.git>
cd age-detect-app
```

7.2.3 Model Download and Installation

You need to download the model file `best.pth` from here.
Next put it in the `age-detect-app/data/checkpoints` directory. Keep the file name as it is. Otherwise, the app will not find it.

7.2.4 Virtual Environment Setup

Navigate to the project directory (`age-detect-app`) and create a virtual environment:

```
python -m venv age_detect_env
```

To activate it use:

`age_detect_env\Scripts\activate` if you are using Windows,
`source venv/bin/activate` for macOS and Linux.

Note 1: On some systems, you might need to use python3 instead of python
Note 2: Your command should now show the virtual environment name

Next, install the required dependencies listed in `requirements.txt` using:
`pip install -r requirements.txt`

7.2.5 Running the Application

Before running the application, ensure you are at the project root directory (`age-detect-app`). Next, type in terminal:

```
python src/main.py
```

The application shall start shortly after this command.

7.3 How to Use

The application has a Qt-based graphical user interface and may look differently on different OSs. After starting the application with 'python src/main.py', a new window will appear. **The gray cursive text in the top left corner gives hints on how to use the interface.** We consider three usage scenarios, each one accessible from the Start menu bar at the top (marked green on Figure 15).

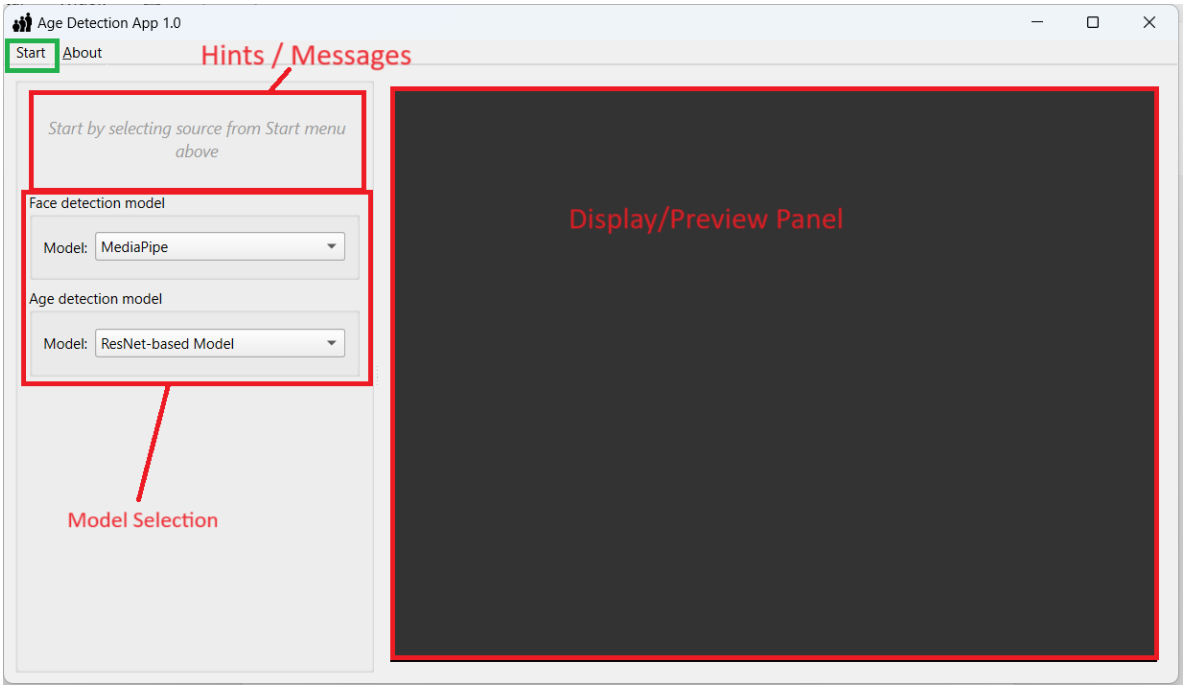

Figure 15: Age Detect App Graphical User Interface

7.3.1 Live Video Mode

To start Live Video Mode, simply click *Start* and *Start Live Capture*. The Preview Panel will start displaying the camera view of the device with live age prediction. Both models can be changed using the comboboxes on the left panel, and the changes will be reflected immediately on the Display Panel. In order to make more accurate predictions, it is advisable to turn on the lights as poor lighting may strongly impact model performance.

7.3.2 Image-Based Mode

To start this mode, you should click *Start* and choose:

- *Select image(s)* if you want to choose only particular image files
- *Select from Folder* if you want to choose and load all images from a specific directory

A file dialog will appear for you to choose the respective files or directory. After accepting your choice, the files will be automatically loaded to the Display Panel in their raw unprocessed form. You may then choose the models and click *Start* button in the left panel to start processing the images (note it is a new button not the menu bar item). When clicked, you should see a progress bar indicating the operation status. Both before and after prediction, you may browse through photos using the ¡, ¿ buttons located below the Display Panel. The gray text in the left panel will guide you throughout the process as always.

7.3.3 Video File Mode

This mode is different to the other ones in that we should start by selecting the models from combo boxes in the left panel. Next, we choose *Start* at the top navigation menu and *Select Video*.

A pop-up window will appear with the disclaimer of chosen models and it will ask the user to confirm the operation. Next, a file dialog will appear prompting the user to choose the video file of interest. Similarly to Image-Based Mode, the user will be shown a progress bar indicating the status of the job. As soon as the processing finishes, a new temporary window will appear with the video modified with bounding boxes and age predictions for people detected. It will close automatically after coming to an end, but it may be also closed manually with no disruption to application.

The software has been tested with `.mp4`, `.avi` formats so far.

7.3.4 Output Files

While output files are displayed in the GUI, they are also automatically saved in `age-detect-app/output` directory. Each request is marked with a timestamp and saved in a subdirectory named after the timestamp within `output`. For output videos, it is recommended to use a robust media player such as VLC to open them correctly.

References

[1] Jingchun Cheng, Yali Li, Jilong Wang, Le Yu, and Shengjin Wang. Exploiting effective facial patches for robust gender recognition. *Tsinghua Science and Technology*, 24(3):333–345, 2019.

[2] Yiming Lin, Jie Shen, Yujiang Wang, and Maja Pantic. Fp-age: Leveraging face parsing attention for facial age estimation in the wild, 2021.

[3] Stylianos Moschoglou, Athanasios Papaioannou, Christos Sagonas, Jiankang Deng, Irene Kotsia, and Stefanos Zafeiriou. Agedb: the first manually collected, in-the-wild age database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshop*, volume 2, page 5, 2017.

[4] Rasmus Rothe, Radu Timofte, and Luc Van Gool. Dex: Deep expectation of apparent age from a single image. In *IEEE International Conference on Computer Vision Workshops (ICCVW)*, December 2015.

[5] Rasmus Rothe, Radu Timofte, and Luc Van Gool. Deep expectation of real and apparent age from a single image without facial landmarks. *International Journal of Computer Vision*, 126(2-4):144–157, 2018.

[6] Song Yang Zhang, Zhifei and Hairong Qi. Age progression/regression by conditional adversarial autoencoder. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. IEEE, 2017.